

Organización del Computador 2

Segundo parcial – 18/06/2019

1 (30)	2 (45)	3 (25)
--------	--------	--------

Normas generales

- Numere las hojas entregadas. Complete en la primera hoja la cantidad total de hojas entregadas.
- Entregue esta hoja junto al examen, la misma no se incluye en la cantidad total de hojas entregadas.
- Está permitido tener los manuales y los apuntes con las listas de instrucciones en el examen. Está prohibido compartir manuales o apuntes entre alumnos durante el examen.
- Cada ejercicio debe realizarse en hojas separadas y numeradas. Debe identificarse cada hoja con nombre, apellido y LU.
- La devolución de los exámenes corregidos es personal. Los pedidos de revisión se realizarán por escrito, antes de retirar el examen corregido del aula.
- Los parciales tienen tres notas: I (Insuficiente): 0 a 59 pts, A- (Aprobado condicional): 60 a 64 pts y A (Aprobado): 65 a 100 pts. No se puede aprobar con A- ambos parciales. Los recuperatorios tienen dos notas: I: 0 a 64 pts y A: 65 a 100 pts.

Ej. 1. (30 puntos)

- (25p) a. Considerando la siguiente tabla de traducciones de direcciones por segmentación y paginación. De ser posible, dar un conjunto de descriptores de segmento, directorio de paginas y tablas de paginas que cumplan con todas las traducciones **simultáneamente**. Detallar los campos de todas las estructuras involucrados. Además indicar desde que segmento de código se esta ejecutando cada acceso, si la traducción es *identity mapping* y en el caso de que alguna traducción no sea posible indicar por qué.

Lógica	Lineal	Física	Características
0x0023:0x000000FF	0x10192FFF	0x00AAEFFF	Datos Nivel 3 Acción: Lectura de 4 bytes, como nivel 0
0x0320:0x00000001	0xFF07A011	0x0391F011	Código Nivel 0 Acción: Lectura de 2 bytes, como nivel 0
0x0323:0x00001233	0xFF07B243	0x003A2243	Código Nivel 0 Acción: Ejecución de 1 bytes, como nivel 3
0x0411:0x00004FFF	0x38442341	0x00333341	Código Nivel 2 Acción: Escritura de 4 bytes, como nivel 0

- (5p) b. ¿Cual es la diferencia entre los bits *accessed* y *dirty* en las entradas de directorios y tablas de paginas? Explicar cómo funcionan ambos bits.

Ej. 2. (45 puntos)

Sea un sistema con segmentación y paginación activa que ejecuta tareas en dos niveles de protección. El sistema ejecuta concurrente 3 tareas de nivel de usuario y una tarea en nivel supervisor. El tamaño de la memoria física es de 1GB y la segmentación debe respetar que el segmento de código de nivel 3 tenga como base 128MB. Cada tarea ocupa 1MB, y el kernel ocupa 2MB.

El sistema provee dos servicios que solo pueden ser ejecutados desde la tarea de nivel supervisor:

- PrendeApagaLaTarea: Toma un identificador de una tarea y la configura de forma que si se encontraba ejecutando, se detenga y viceversa.
- ModificarRegistros: Toma un identificador de tarea, un identificador de un registro y un valor, y carga el registro indicado con el valor pasado por parámetro, para la tarea ejecutando en nivel 3.

- a. Definir un posible mapa de memoria para las tareas. Completar el mapa con el rango de direcciones de las paginas utilizadas para el kernel y cada una de las tareas. Indicar el contenido de base, límite

y tipo de los descriptores de GDT para datos y código. Indicar el rango de traducciones de linear a físicas propuesto para paginación. Definir las entradas de la IDT para los servicios pedidos, para las rutinas de excepciones y para la interrupción de reloj.

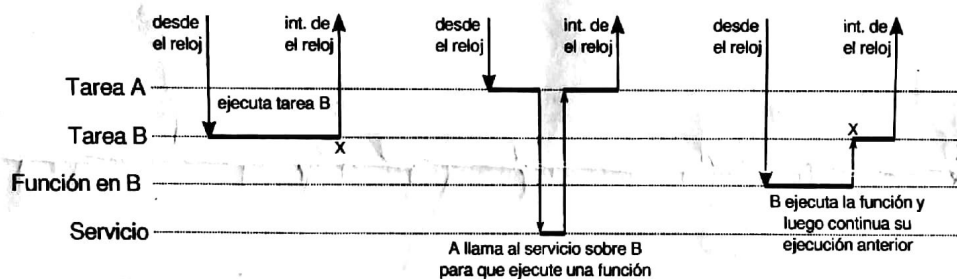
(15p) b. Implementar en ASM y/o C la rutina de atención de interrupciones del reloj y el servicio **PrendeApagaLaTarea**.

(15p) c. Implementar en ASM y/o C la rutina de atención de interrupciones del servicio **ModificarRegistros**.

Nota: Suponer que se cuenta con las definiciones de las estructuras del sistema para operar desde C.

Ej. 3. (25 puntos)

Se desea implementar un servicio por el cual una tarea A le indique a otra tarea B que ejecute una función en su espacio de ejecución y luego continúe con su ejecución como si nada hubiera pasado. Para esto se supone que A conoce la dirección lógica de la función en el espacio de ejecución de B, por lo que el servicio debe tomar como parámetros dicha dirección y el identificador de la tarea B. La función por su parte debe correr en el contexto de B con nivel usuario, tomando como parámetros la dirección de retorno donde la tarea estaba ejecutando originalmente y el valor de sus flags, para que luego de ejecutar la función pueda retornar a su estado original.



La figura anterior ilustra tres eventos. La tarea identificada como B ejecutando y desalojada al momento de llegar a x. La tarea A llamando al servicio, indicando que la tarea B ejecute la función f. Y por último, la tarea B ejecutando nuevamente pero en esta ocasión ejecutando la función f para luego continuar la ejecución desde x.

- (5p) a. Explicar el funcionamiento del servicio pedido y que condiciones considera sobre el sistema.
- (20p) b. Implementar la rutina del servicio pedido.

Considerar que se proveen una función, que dado un id identificador de tarea, retorna el puntero a la tss (`tss* getTss(uint32_t id)`).

Además la aridad de la función a llamar debe respetar: `void f(uint8_t flags, void* returnAddress)`

Nota: De ser necesario puede utilizar las instrucciones `lahf` para cargar flags o `sahf` para leer flags.